ESEIAAT
TREBALL DE FI DE GRAU

GRAU EN ENGINYERIA EN TECNOLOGIES INDUSTRIALS

ESTUDI, DISSENY I IMPLEMENTACIÓ D'UN HUMANOIDE PEL SEGUIMENT DE TRAJECTÒRIES

ALUMNE: FERRIOL PEY COMAS
DIRECTOR: JOSEBA QUEVEDO CASIN
ENTREGA: 10 DE JUNY DE 2019

Dedicat a ma mare i a mon pare en agraïment
a la seva infinita paciència i dedicació en tot
el que han fet per mi.

Mama, papa, aquest és el treball del vostre petit inventor

Índex

Índex de taules

Índex de figures

Introducció

1.1 Resum

Els humanoides són robots que posseeixen una configuració similar a la morfologia humana, és a dir, que consten d'un cap, un tronc, dos braços i dues cames (en certs casos els humanoides poden simular-ne sols una part, anomenant-se llavors humanoides parcials). La forma més habitual de locomoció dels humanoides és caminant, i és per això que presenten especial dificultat en el món de la robòtica.

La intenció d'aquest estudi és la de dissenyar, construir i programar un humanoide capaç de seguir trajectòries. Per a tals efectes, el robot (batejat com a Tamtam) haurà de ser capaç de desplaçar-se de forma estable seguint la ruta que ell mateix haurà disposat. Així doncs es plantegen tres objectius: el primer és el de ser estable, el segon implica que ha de capaç de caminar seguint un rumb definit i el tercer és el de calcular el traçat que s'ha de seguir per arribar al punt desitjat sortejant, si s'escau, els obstacles que ho impedeixin.

Per a assolir l'objectiu de la locomoció es proposa l'estudi de l'estabilitat de l'humanoide i el control de la mateixa utilitzant la tècnica de control PID. Després es pretén aplicar *zero moment point* per aconseguir el desplaçament. Pel que fa al traçat de trajectòries, s'usa les llibreries d'OpenCV i ZBar amb una càmera per trobar els codis QR que identifiquen els obstacles i punt d'arribada i la seva ubicació per definir el trajecte a seguir. Un cop aconseguit resta la implementació del seguiment de la trajectòria trobada.

Amb la tecnologia actual i les tècniques de control que existeixen, és possible la materialització d'un humanoide com el descrit en aquest projecte. S'ha aconseguit la seva estabilització, cosa que fa que sigui més fàcil el desplaçament caminant. També s'ha aconseguit la detecció dels marcadors. Tanmateix és una feina laboriosa. Cada vegada hi ha més informació sobre

el tema, i també molts més projectes que apliquen tècniques noves. El problema ara es troba en la implementació, ja que la informació disponible està encarada a projectes amb característiques molt diferents a la del projecte, per tant l'adaptació s'ha hagut de cuidar molt. De cara a la continuació d'aquest projecte es podria buscar un entorn més semblant al de la competició CEABOT i buscar que el robot aprofiti parts d'aquest projecte per assolir-ne més proves. Aquest concurs és un concurs organitzat per al grup CEA (Comitè Espanyol d'Automàtica) que es realitza cada any des del 2006 entre universitaris i grups de recerca de tota Espanya. En la prova hi ha un camp de 2x2.5m amb les parets marcades amb QR i una sèrie d'obstacles també marcats (el competidor en desconeix la quantitat i la disposició). L'objectiu de la prova és que l'humanoide travessi el camp i després torni al punt de partida en el menor temps possible (penalitzant caigudes i xocs amb els obstacles). Fins ara ningú no ha realitzat aquesta prova utilitzant visió, sinó que ho han fet amb sensors òptics. És per això que aquest problema és interessant, i fins i tot té una aplicació directe en el món de la competició.

1.2 Declaració d'honor

Jo declaro que,
el contingut d'aquest treball de fi de grau és de la meva autoria, no s'ha agafat res de treballs de ningú sense esmentar-los i donar-los el crèdit que pertoca, totes les referencies han estat degudament citades, i entenc que la infracció d'aquesta declaració queda subjecte a les mesures disciplinaries que s'escaiguin per part de la Universitat Politècnica de Catalunya - BarcelonaTECH.

Ferriol Pey Comas 09-06-2019

1.3 Objecte

L'objecte d'aquest treball és el de dissenyar, construir i programar un prototip d'humanoide que tingui la capacitat de traçar trajectòries i posteriorment seguir-les desplaçant-se caminant de forma estable. Així doncs l'estabilitat, la locomoció i el càlcul de trajectòria són d'interès en aquest projecte.

Es pretén també buscar els components, programari i tècniques que més bé s'adaptin al projecte (sota criteris de cost, qualitat i simplicitat en aplicació) per tal d'aconseguir la màxima viabilitat del mateix.

1.4 Abast

Entra dins l'abast del projecte el disseny de cadascuna de les peces impreses que conformen l'humanoide així com de la seva Impressió, el disseny i soldadura de la placa auxiliar de control, el disseny i programació del conjunt de programes necessaris per fer que el robot funcioni, el disseny i implementació dels controladors PID que s'escaiguin per a la correcta estabilització, l'estudi del control de trajectòries i de la seva materialització, l'assemblatge del robot i la discussió dels resultats i futurs.

No entra dins l'abast del projecte la fabricació de la placa dissenyada ni els càlculs estructurals de les peces impreses de l'humanoide.

1.5 Requeriments i especificacions

Els recursos necessaris per a dur a terme el projecte s'exposaran a continuació, juntament amb una breu descripció i justificació:

Kicad és un software lliure (sota llicencia GNU) de disseny de plaques electròniques. S'ha triat aquest programa per la potència que té, perquè està disponible als repositoris de pràcticament tots els sistemes operatius en base Linux i perquè és lliure.

OpenSCAD és un programa de disseny 3D lliure que funciona renderitzant un arxiu de codi font que conté un conjunt d'elements bàsics (com ara esferes, cilindres o cubs..), unes operacions espacials (translacions, rotacions, escalats...) i unes operacions binàries (diferencia, unió, intersecció) que combinades conformen el model. També inclouen instruccions pròpies dels llenguatges de programació com ara condicionals, iteratius i creació de funcions (mòduls) que en fan molt fàcil l'ús un cop familiaritzat. S'ha escollit per la seva versatilitat i la facilitat de realitzar modificacions en peces (és molt senzill fer-les paramètriques), la gran capacitat de reaprofitar codi i també per tenir llicencia lliure.

Matlab matlab és un llenguatge de programació i un entorn de desenvolupament que en aquest projecte s'usarà per els mòduls que té per trobar la funció de transferència d'un sistema i poder simular el comportament del mateix amb realimentació i sistemes de control. No és gratuït, però degut a que aquest projecte és per a fins educatius s'usarà la llicència d'estudiants que es cedeix a la UPC.

Eines necessàries per al muntatge del robot com ara soldador per tal de soldar el cablejat i els components a la placa, tornavís per a poder cargolar els cargols necessaris per tal de fixar les peces, cutter per a repassar les peces impreses, fundes termoretràctils per protegir les unions entre cables, cinta de doble cara per fiançar les unions entre dues peces que hagin de ser solidàries...

Raspberry Pi La raspberry pi és un ordinador de la mida de una targeta de crèdit basat en tecnologia ARM, i consta de diferents models, en aquest projecte s'usarà la **Zero W** degut a que encara té una mida més reduïda i porta targeta wifi integrada. S'ha escollit la Raspberry Pi per tenir una àmplia documentació, disposar de molts recursos en programari, i tenir un cost molt assequible. El sistema operatiu emparat és Raspbian (basat en Debian) en la seva versió lite.

Ordinador amb un sistema Linux instal·lat (en aquest cas s'usarà un *Arch Linux*). S'ha escollit una distribució de Linux per la facilitat de configurar i instal·lar programes lliures (Kicad , openscad i ROS per exemple) i per comptar amb eines com ara ssh que estableixen la connexió amb la raspberry

pi.

Xarxa wifi en la que s'hi trobin l'ordinador i la raspberry per tal de fer possible la comunicació. S'ha optat per usar el mòbil com a router per tenir més comoditat a l'hora de sortir de la zona de treball habitual i fer més portable el sistema.

OPENCV, ZBar, ROS, g++, python Tots ells programes i llibreries lliures que permeten la funcionalitat completa dels programes del projecte. S'han escollit per la seva versatilitat, documentació i facilitat d'implementació.

1.6 Justificació

Els humanoides sempre han presentat una dificultat afegida en el camp de la robòtica degut als problemes d'estabilitat. És per això que el desplaçament d'un humanoide que es mou caminant és complex.
Actualment hi ha una sèrie de competicions de robòtica que centren la seva atenció en humanoides. Aquestes competicions posen a prova la capacitat que tenen aquests humanoides de relacionar-se amb l'entorn (ja sigui captant-ne informació i desplaçar-se sense caure o interactuant directament amb el medi manipulant-ne algun element).
Aquest projecte pretén abordar una simplificació del repte que proposa el concurs CEABOT.

Descripció de l'humanoide

Aquest humanoide està pensat per a poder resoldre una de les proves del CEABOT, concretament la del laberint. Aquesta prova consta d'un camp rectangular de 200cm x 250cm amb parets de 50cm d'altura, i està disposat seguint la cara llarga . El terra és verd fosc, però compta amb dues línies blanques, una marca la línia de sortida i la altre la línia parcial d'arribada. Aquestes línies es troben separades 50 cm de la paret, de tal forma que queden 150 cm entre elles. Les parets estan marcades amb codis QR cada 50 cm que informen de la paret on estan (N per la nord, S per la sud E per la est i W per la oest) i de la distància en el seu eix (començant per a 25 i acabant en 225, així doncs, un marcador posat a la paret est a la meitat del camp contindria la cadena de caràcters *E125*). Dins el camp s'hi troben disposats obstacles (fins a un màxim de 6) que tenen una base quadrada de 20cm de costat i una altura de 50cm. Aquests seran inamovibles en la mesura del possible i també tindran un marcador QR que indicarà el nombre de l'obstacle i la cara en la que està disposat. Els obstacles seran col·locats pel jurat un cop tots els participants hagin deixat el robot a la taula del jurat, fent impossible que es modifiqui el codi un cop sabuda la seva disposició. L'objectiu de la prova és arribar a la línia d'arribada parcial, creuar-la i tornar a la zona de sortida en el menor temps possible, i sense caure ni xocar amb els obstacles. Fins a dia d'avui cap dels participants ha fet us de la visió per resoldre el problema que presenta aquesta prova, així doncs aquest projecte pretén donar les eines per a poder-lo realitzar. El problema tractat serà una versió simplificada del mateix, però extrapolable perfectament al cas del laberint.
A més a més aquest humanoide està pensat per ésser a compartit amb la comunitat *open source*.

2.1 Antecedents

Un robot pot ser un programa que usa internet per a realitzar una tasca molt concreta o també pot ser una màquina programable que per la seva complexitat permet adaptar-se a diverses tasques. Els robots tenen un ampli us, des de la indústria a la recerca. La tradició dicta tres normes principals imposades per un escriptor rus anomenat Isaac Asimov, que escrivia novel·les de ciència ficció on els seus robots sempre les seguien. Aquestes normes són:

1. Un robot no podrà sota cap concepte ferir a un ésser humà, o per inacció permetre que passi.
2. Un robot ha de seguir les ordres dels éssers humans a no ser que aquestes violin la primera llei.
3. Un robot ha de vetllar per la seva protecció sempre i quan no entri en conflicte amb les dues lleis anteriors.

Aquestes normes sols són un referent, ja que tots els robots necessiten d'un o diversos programes per funcionar, per tant, són les persones les que marquen el seu comportament. Tot i així a vegades s'obté algun comportament no desitjat. Per exemple el 23 de març de 2019 [11] Microsoft es va aventurar en un viatge que no arribaria a bon port, va presentar al món un nou robot anomenat Tay. Aquest era un robot basat en intel·ligència artificial que tenia un compte a *Twitter* i interactuava amb els altres usuaris. Aprenia de les converses que tenia amb ells i anava "incorporant a la seva personalitat" tot el que aprenia. Doncs bé, en menys de 16 hores aquest robot va ser descartat degut a que va adoptar una ideologia nazi, xenòfoba i sexista, cosa que va agafar per sorpresa als seus creadors, que mai havien pensat que això podria ocórrer.

Un altre cas bastant famós és el de dos robots de Facebook pensats per aprendre a negociar amb humans. Aquests robots es van fer negociar entre ells per una sèrie d'objectes que hi havia en una matriu, i es van deixar aprendre l'un de l'altre. El que esperaven els creadors d'aquest curiós experiment era la creació d'uns robots capaços de negociar amb humans per internet d'una forma tant fluida i natural que els humans amb els que interactuessin fossin incapaços de detectar que a l'altre extrem de la comunicació no hi havia cap persona, sinó un robot. El resultat però va ser ben diferent, el que es va aconseguir va ser que els dos robots desenvolupessin un nou llenguatge per a comunicar-se que era inintel·ligible per als humans a simple vista, així

que l'experiment va ser descartat.

Casos com aquest fan pensar que barrejar la intel·ligència artificial amb la robòtica pot ser perillós, i que per tant estaria bé la implementació d'aquestes tres lleis esmentades anteriorment. Hi ha obres narratives que en narren les possibles repercussions, o fins i tot hi ha pel·lícules que mostren desastres tot i haver-hi les tres lleis en joc. Aquestes obres són òbviament ciència-ficció i estan molt lluny de ser certes, tanmateix si que plantegen idees interessants i debats morals sobre el que s'ha de poder fer i el que no, i recordant que un petit error en la programació pot ser fatal per al projecte (ja que el robot pren un curs completament imprevist) ens fa preguntar si realment és segur jugar amb aquesta tecnologia desconeguda.

Hi ha moltes formes de classificar els robots, però en aquest projecte sols es centrarà l'atenció en els robots humanoides. Aquests són aquells que tenen una morfologia humana, per tant s'assemblen a les persones. Els humanoides que sols simulen una part de la morfologia humana es diuen humanoides parcials. Els androides són un cas particular dels humanoides, mentre que els humanoides sols imiten la forma del cos humà els androides tenen la intenció de reproduir-lo amb la màxima fidelitat possible. Tant és així que fins i tot simulen expressions i gestos humans.

Els humanoides tenen dos camps d'actuació molt definits, el primer és el sector dels serveis, i el segon és el món de la recerca.
Pel que fa al primer, és lògic pensar que quan una persona ha de fer ús d'algun servei es senti més còmode amb un robot amb aparença humana que no amb una màquina que intimidi, el fet de personificar un robot genera confiança en els usuaris, i al tenir una forma similar a les persones amb qui interactuen fa que no presentin tant rebuig.

Asimo (figura 2.1) és un robot de l'empresa Honda que clarament mostra aquesta ideologia. És un humanoide que pesa 48kg i que mesura 130cm, pot caminar, córrer i fins i tot saltar a peu coix. Compta amb 57 graus de llibertat i és capaç de desenroscar el tap d'una ampolla i emplenar un got. Aquest humanoide ha estat pensat per interactuar directament amb persones, per tant s'ha buscat que compleixi amb una serie de característiques, com ara que té un aspecte amigable, genera confiança degut a la seva aparença indefensa, té una veu suau, els seus moviments són fluids i naturals.

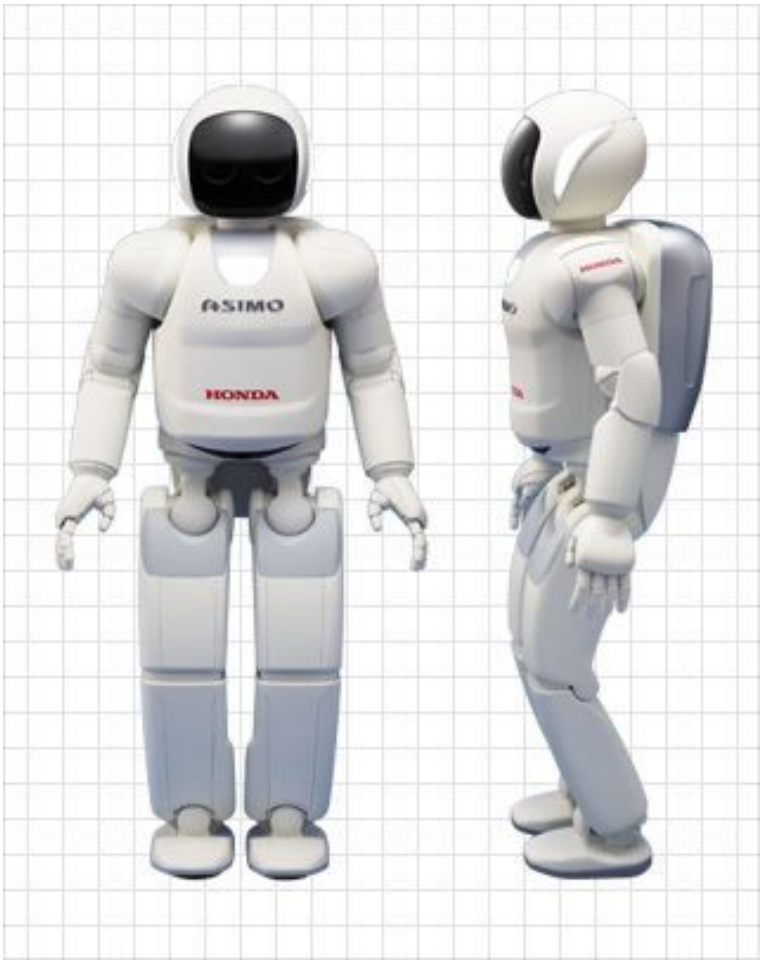


Figura 2.1: Humanoide asimo
[7]

Per altre banda, Atlas (figura 2.2) és un humanoide creat per Boston Dynamics, una empresa de robòtica nord-americana. Aquest humanoide no és tant suau estèticament, però en canvi té unes prestacions diferents. Medeix150cm i pesa 75Kg, aquest humanoide pot caminar, córrer, saltar i fins i tot fer salts mortals. Aquest humanoide és més aviat per al món de la recerca, en el qual la part estètica no és tant valorada com la seva funcionalitat.

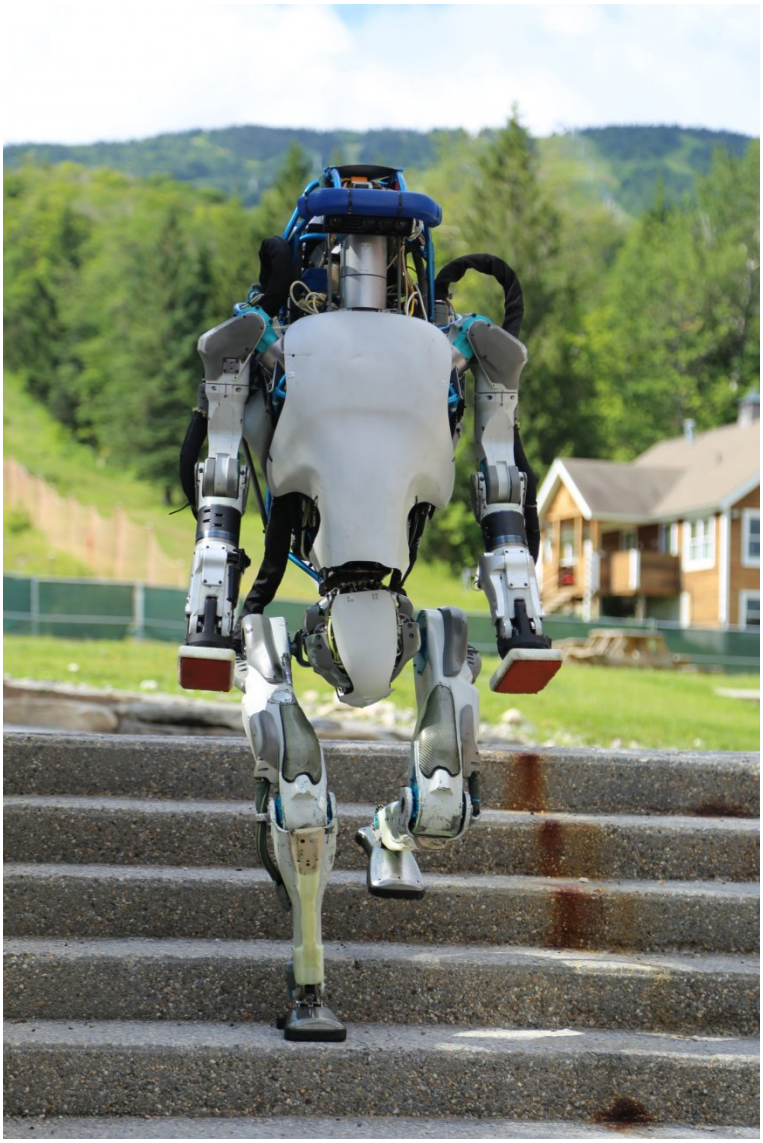


Figura 2.2: Humanoide atlas
[1]

2.2 Disseny proposat

Per a poder començar a dissenyar l'estructura del robot s'ha de tenir un punt de partida, unes directrius que defineixin de forma orientativa com haurà de ser. Amb això en ment es definiran els següents aspectes: es planteja un humanoide de mida reduïda (entre 20 i 25 cm d'altura), cosa que farà molt

més simple el seu control, més econòmic ja que al ser més petit pesarà menys i tindrà menys longitud entre articulacions, cosa que reduirà considerablement el parell necessari per al seu control, i per tant, uns actuadors menys costosos, i també perquè les peces seran més petites, el que deriva en una menor despesa en plàstic i hores d'impressió. També s'estipula que el cos del robot haurà d'aproximar en la mesura del possible el canon que presenta el cos humà. Una altre propietat desitjable però no determinant és aconseguir un pes final de l'ordre de 600g, cosa que donarà més agilitat a l'humanoide. Amb 19 DOF (*degrees of freedom* o graus de llibertat en català) es tenen les articulacions principals del cos humà, i farà que el robot tingui una alta llibertat de mobilitat, permetent que sigui més fàcil la imitació de gestos similars als dels humans.

Per a poder dissenyar, construir i programar l'humanoide, abans s'ha de conèixer les principals problemàtiques que hi ha en la seva materialització així com les seves possibles solucions. A continuació s'exposen els principals punts a discutir i la solució o solucions proposades així com una breu justificació de les mateixes.

2.2.1 Articulacions

Entenem com a articulació tots aquells punts d'unió mòbil entre dues peces del robot dels quals el moviment permès és el de rotació. Les articulacions solen comptar amb actuadors, que són els que generen el moviment que s'hi produeix. La conjugació dels moviments en les articulacions resultaran en el moviment del robot, com a conseqüència els actuadors han de tenir la força necessària per a poder suportar el pes del robot. També han de ser prou ràpids com per mostrar un moviment fluid i natural per evitar cops bruscs en el moment de generar el moviment que es desitja. Així doncs, el problema de les articulacions queda reduït a la selecció dels actuadors més adequats per al projecte. És important recordar també que s'ha de tenir en compte que els actuadors han de tenir unes mides compatibles amb les peces en les quals s'han d'ubicar.

L'actuador seleccionat per aquesta aplicació s'ha escollit basant-se en la mida, efectivitat, cost i simplicitat d'implementació. Com a resultat s'ha escollit el servomotor. El motor DC amb encoders i reductora era una alternativa, però planteja problemes com ara la gestió de cada un de les articulacions tant a nivell mecànic com de potencia i control, cosa que fa la seva electrònica i programació molt més complicada. També s'ha considerat

el motor pas a pas, però tot i tenir una alta precisió te molt baix parell, i planteja problemes en les seves posicions inicials (quan el robot s'encén se'n desconeix la posició, per tant es fan servir finals de cursa per fer-lo anar fins al punt inicial). A més a més, els drivers necessaris per a poder controlar les 19 articulacions ocuparien molt volum, cosa que no és compatible amb l'especificació d'un humanoide de mida reduïda. Així doncs s'ha optat per els servomotors.

Els servomotors són uns actuadors que posicionen l'eix de sortida a un angle concret escollit per l'usuari i seleccionat mitjançant una senyal PWM (*pulse width modulation* o modulació de l'amplada de pols).Aquest angle tradicionalment comprenia dels 0 graus fins als 180, però hi ha servomotors que permeten un rang de 0 a 270 graus o, fins i tot, de 0 a 360 graus. L'amplada del pols del servomotor és de 20 ms, inicialment s'envia un pols lògic alt de 1ms a 2ms, on 1ms correspon a 0 graus i 2ms correspon a 180 graus, i presenta un comportament lineal, així doncs per posicionar-lo a 90 graus se l'hi envia un pols de 1,5ms. De l'actuador en surten tres cables, un de Vcc (és l'entrada de voltatge positiva al motor), un de GND (el negatiu de control i del motor) i un de senyal (en el que s'envia el PWM que controla l'angle al qual es vol que es posicioni).

Estan constituïts per un circuit de control, el qual calcula la posició actual de l'eix a partir d'un potenciòmetre (resistència variable) ubicat a l'eix de sortida i la direcció en la que ha de girar el motor i a quina velocitat (corresponent baixes velocitats a variacions d'angles petites i altes velocitats a variacions d'angles més alts). El càlcul de la velocitat del motor es fa amb un PD, així que si l'eix presenta alguna resistència no arribarà a la posició desitjada. L'element de potencia és un motor elèctric de corrent continu, que rep potencia en forma de PWM (seria un símil a regular-ne el voltatge d'entrada però de forma digital), i aquest transforma l'energia elèctrica en energia mecànica que transmet a la caixa de reducció. A la caixa de reducció hi ha una serie d'engranatges que redueixen la seva velocitat però n'augmenten la força, i a la sortida d'aquesta caixa de reducció s'hi ubica l'eix de sortida. Aquests engranatges solen ser de plàstic per als servomotors més senzills, mentre que per als més exigents són metàl·lics.

Degut a les restriccions obtingudes s'ha arribat a dos possibles models:

Els actuadors escollits han estat els MG90S, perquè tot i ser més cars tenen millors prestacions.

Model	Parell	Mides	Velocitat	preu
SG90	1,5Kg/cm	22,0 x 11,5 x 27,0 mm	0,3s/ 60°	1,35 €/u
MG90S	2,2Kg/cm	22,8 x 12,2 x 28,5 mm	0,08s/ 60°	1,83 €/u

Taula 2.1: Taula comparativa dels servomotors

2.2.2 Control

Un cop seleccionats els actuadors que s'usaran s'ha de trobar la millor forma de controlar-los així com a la resta de components del robot. El més habitual en aquests casos és fer servir un servomotor driver, que és un element que genera les senyals PWM dels servomotors que és capaç de suportar (el nombre màxim de servomotors que pot controlar és indicat com a numero de canals, així doncs si té 23 canals pot suportar fins a 23 servomotors). Aquests elements es solen controlar per I2C (IIC, *inter integrated circuit*) que és un sistema de transmissió de dades bidireccional i síncron que sols necessita de tres cables. El primer és l'SCL (*serial clock*), és el que sincronitza les comunicacions, és una senyal de rellotge i és generat per el mestre del bus. El segon és l'SDA (*serial data*), i és el que transmet la informació, que pot procedir del mestre o de qualsevol dels esclaus. Per acabar, el tercer cable és el GND i actua com a referència.

Un dels avantatges del I2C és que té un dispositiu definit com a mestre, però pot tenir fins a 112 dispositius esclaus sempre i quan no es repeteixin les adreces. Les adreces són de 7 bits (tot i que l'estàndard també defineix modes de direccionament de 8, 10 i fins i tot 12 bits), cosa que dona un total de 128 adreces diferents, però 16 d'aquestes 128 estan reservades. Com que la unitat de transmissió són paquets de 8 bits (8 bits equivalen a un byte) i les adreces són de 7 bits, es fa servir el bit restant com a indicador de lectura/escriptura (si el 8é bit és igual a 1, la operació és de lectura, si és igual a 0 és d'escriptura). A més a més fa moltes comprovacions per assegurar que el missatge ha arribat d'un dispositiu a l'altre.

Un dels desavantatges més greus és que és lent degut a totes les comprovacions que realitza. Un altre dels desavantatges és que la connexió sempre la iniciarà el mestre, un esclau mai podrà fer-ho. De la mateixa forma és impossible que un esclau es comuniqui amb un altre esclau.

El *servo driver* escollit és l'SD20, és un microcontrolador (PIC16F872) programat per enviar les senyals de control de 20 canals i controlat per I2C,

corre a 8 MHz i té un funcionament molt simple.

Com a controlador principal s'ha escollit la *Raspberry Pi Zero W* ja que és un ordinador amb molta documentació, amb 28 GPIO (entrades/sortides de propòsit general), compatible amb la majoria d'eines que hi ha per Linux, porta targeta de xarxa wifi integrada i consumeix poc. Un dels problemes més greus que tenen aquests ordinadors és que no compten amb cap entrada analògica, són totes digitals.

Entenem com a senyal digital (per a aquest document serà un sinònim de senyal digital binaria) una senyal que pot prendre dos valors diferents: 0 lògic (tots aquells voltatges que queden per sota d'un voltatge definit com a voltatge llindar) o 1 lògic (tots aquells voltatges que superen el valor del voltatge llindar). Així doncs aquestes entrades o sortides sols poden tractar estats binaris.

Entenem com a entrada analògica (en aquest projecte no hi haurà cap dispositiu que compti amb sortides analògiques) tota aquella entrada que en arribar-li un voltatge és capaç d'assignar-li un valor que dependrà del seu voltatge màxim i de la resolució del dispositiu que llegeix. Per exemple, si el voltatge màxim és 5V i la resolució són 10 bits, el valor 1023 es correspondrà amb 5V, el valor 0 amb 0V, i el 511 amb 2,5V. El voltatge d'entrada quedarà discretitzat en graons de 0,004887V (corresponents a 5V/1023), i d'aquesta forma es pot obtenir un rang més extens que els digitals, tanmateix és més lent degut al procediment que usa per calcular el valor.

És per aquest motiu que es proposa l'ús d'un microcontrolador auxiliar que tregui càrrega a la Raspberry i que també serveixi com a extensió per tenir entrades analògiques. Així doncs es proposa l'ús de l'ATMEGA328P-AU, que és una versió SMD del microcontrolador ATMEGA328P de la marca AVR. S'ha escollit perquè té una mida molt reduïda, compta amb 8 entrades analògiques (de les quals dues es destinaran a la comunicació I2C i una al control de la bateria) i 13 digitals (de les quals tres es destinaran al control de leds RGB (leds dels quals se'n pot definir el color) que s'ubicaran als ulls per usar com a indicadors o fins i tot com a sistema d'interacció). A més es pot programar en Arduino, cosa que simplifica la seva implementació.

Per acabar, també és necessari l'ús de sensors per a poder captar informació de l'entorn en el qual es troba. Per a poder controlar les inclinacions i rotacions que presenta l'humanoide es proposa l'ús d'una IMU (*inertial mesurement unit* o unitat de mesura inercial), que és un sensor que capta

informació referent a velocitats angulars i acceleracions del propi dispositiu. Així doncs, una de les millors IMU's que hi ha al mercat és la BNO055. Aquesta IMU no sols calcula acceleració i velocitat angular en cada un dels eixos, sinó que també porta magnetòmetre (és com una brúixola electrònica, capta variacions de camp magnètic per determinar el pol nord terrestre), i porta un microcontrolador que aplica un filtre de kalman a les senyals per estimar-ne la orientació respecte cadascun dels eixos. Un filtre de kalman és un algoritme que coneguen les variàncies de cadascuna de les senyals les fusiona per obtenir una senyal de sortida amb menys error, és a dir, estima una variable amb molt poc error a partir de mesures amb error de la pròpia variable.

La IMU permetrà que el robot conegui en tot moment la seva inclinació per a poder-la corregir i també la seva orientació (rotació absoluta en l'eix Z), que servirà per corregir la trajectòria que ha de seguir.

Per altre banda la IMU només permet obtenir informació referent al robot. Així doncs necessitem d'altres sensors que dotin el robot de la capacitat de conèixer sobre l'ambient en el que es troba. Una bona forma de fer-ho és amb visió artificial, és a dir, a través del tractament d'imatges obtingudes a partir d'una càmera que portarà el mateix robot a sobre. S'ha de decidir on s'ubicarà per tal de permetre'n el màxim profit, així doncs es proposa al cap, ja que pot rotar i ,per tant, dona la possibilitat d'orientar la càmera cap on sigui necessari. Per tal de fer més simple la seva implementació també es poden marcar amb codis QR els elements d'interacció més importants, disminuint així la complexitat del problema. Utilitzant aquest tipus de sensors es poden ubicar una serie d'elements identificats per color, forma o marcador a l'espai utilitzant un sistema de referència solidari al robot. Així doncs amb aquesta tècnica l'humanoide serà capaç de calcular on està el punt d'arribada i on es troben els obstacles.

El control de l'humanoide també fa referència al sistema d'actuadors i a la forma de controlar-los. Es per això que es presenta un esquema(figura 2.3) del funcionament del mateix així com una descripció de cada un.

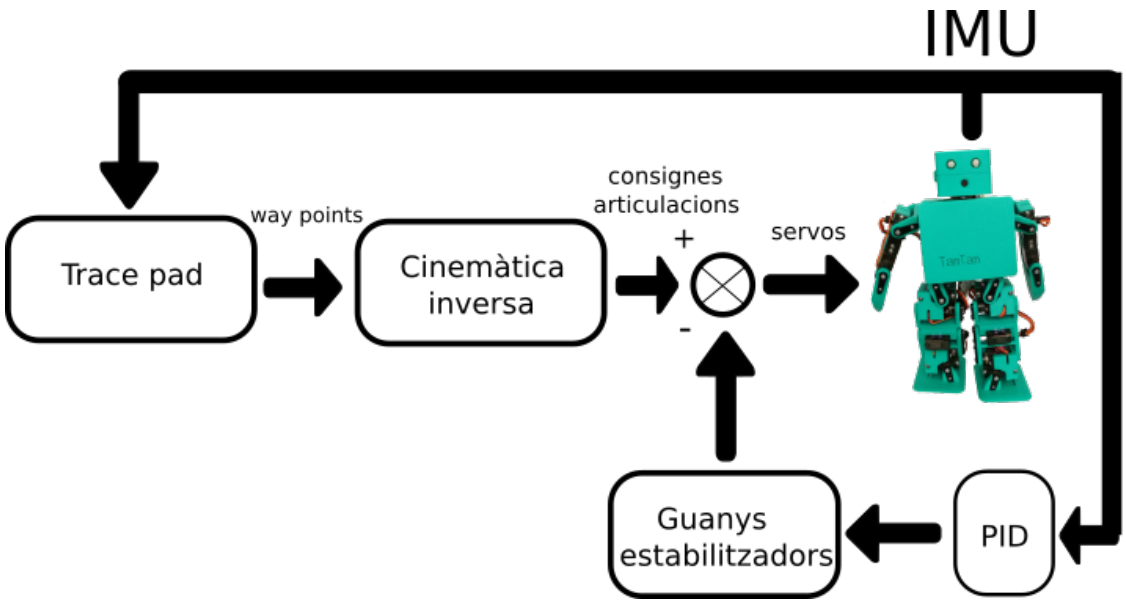


Figura 2.3: Esquema del control d'en Tamtam

Servos, els servomotors porten un controlador P intern, en aquest projecte els servomotors seran com una caixa negra. Així doncs, no es tocarà res del control intrínsec del servomotor.

PID, és un controlador que consta de tres parts. La primera part és la P (és un regulador proporcional), dona una resposta proporcional a l'error que hi ha. El segon bloc és el D (que és un regulador derivatiu) i bàsicament el que fa és estudiar com varia l'error i dona una resposta que depèn de la variació de l'error en el temps. D'aquesta manera s'aconsegueix que la resposta s'estabilitzi de forma més ràpida. Per últim, el bloc I (que és un regulador integral), és un bloc que fa que l'error tendeixi a zero. En la majoria de sistemes, en un PD s'arriba a l'estabilitat sense arribar a la consigna desitjada, per això es fica el bloc I. Aquest bloc acumula l'error obtingut i el multiplica per una constant, fent així que encara que el sistema PD arribi a l'equilibri, si hi ha error es pugui corregir.
El PID es pot definir de forma temporal com a:

$$u(t) = Kpe(t) + Ki\int_0^t e(t)dt + Kd\frac{de}{dt}$$

On *u(t)* és la sortida del PID i *e(t)* és l'error en funció del temps. El

problema d'aquesta expressió és que defineix el PID de forma contínua, així doncs s'ha de discretitzar per a poder adaptar-lo al projecte. L'adaptació proposada és la següent:

$$u(n) = Kpe(n) + KiTm\Sigma_{i=0}^{n-1}e(i) + Kd\frac{e(n) - e(n-1)}{Tm}$$

On Tm és el període de mostreig i n és l'instant actual de càlcul. Per evitar que el sumatori sigui més llarg cada vegada es pot definir el càlcul precedent com a :

$$u(n-1) = Kpe(n-1) + KiTm\Sigma_{i=0}^{n-2}e(i) + Kd\frac{e(n-1) - e(n-2)}{Tm}$$

I al restar el càlcul anterior a l'actual es pot definir una expressió que sols depengui del valor actual i de l'anterior, i així evitar fer un sumatori que s'estén a cada període. L'expressió és:

$$u(n) = u(n-1)+Kp(e(n)-e(n-1))+KiTme(n-1)+Kd\frac{e(n) - 2e(n-1) + e(n-2)}{Tm}$$

El PID que portarà el robot servirà per estabilitzar-lo. Per aconseguir-ho s'ha realitzat un experiment per a recol·lectar informació per a trobar el model que identifica el sistema del robot i que té com a entrades l'angle δ (explicat a la cinemàtica de l'humanoide) i com a sortida el *pitch* de l'humanoide (angle respecte l'eix Y) (figura 2.6). Així doncs, s'ha fet un programa que llegeix el valor del *pitch* provocat per un graó d'un angle δ de 20°, i guardava en l'arxiu el valor del *pitch* i el valor de l'instant de temps en que s'ha pres la mesura. Un cop aquí s'ha introduït les dades a un mòdul de matlab anomenat ident (pertanyent a la toolbox *System Identification Toolbox*). Aquesta toolbox busca models matemàtics o dinàmics a partir de dades de senyals d'entrada i sortida que no són fàcils de de modelar. Permet l'ús del domini temporal o del domini freqüencial per a identificar funcions de transferència continues o discretes en el temps. També inclou sistemes que estimen de forma automàtica els seus paràmetres [10].

S'ha configurat de tal forma que estimi la funció de transferència del sistema amb un retard pur , dos pols i cap zero, i s'ha obtingut la funció de transferència següent:

$$G(s) = \frac{194,1}{s^2 + 12,91s + 121,9}e^{-0,039s}$$

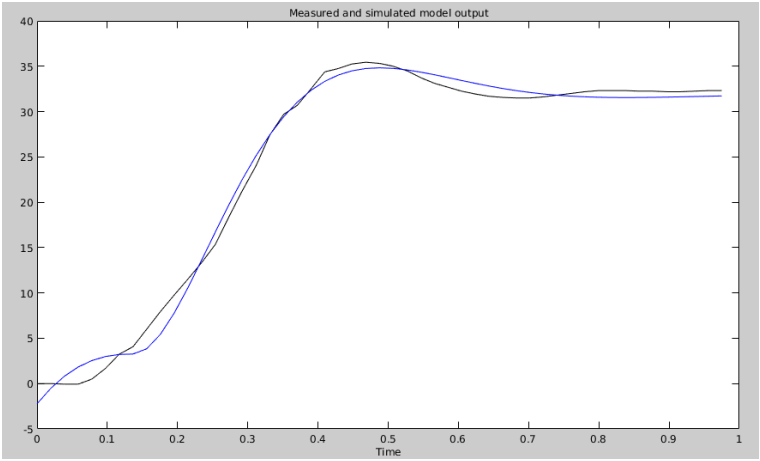


Figura 2.4: Gràfica comparativa de la resposta real amb la resposta del model calculat

En aquest punt s'ha usat el mètode de Philips [6] per a calcular un PID que sobre el paper dona aquests resultats amb aquestes constats:

Kp	**Ki**	**Kd**
1,1	4,0	0,102

Taula 2.2: Taula de les constants del PID del model inicial

Figura 2.5: Gràfica teòrica del PID inicial

Un sistema que arriba a l'equilibri en menys de mig segon i de forma molt suau. Però en la seva implementació el sistema és inestable, així que s'han agafat les dades de la sortida del PID i la resposta del sistema (pitch) per a generar un model nou més ric. S'ha configurat de tal forma que estimi la funció de transferència del sistema amb un retard pur (que ha resultat ser inexistent), dos pols i un zero, i s'ha obtingut la funció de transferència següent:

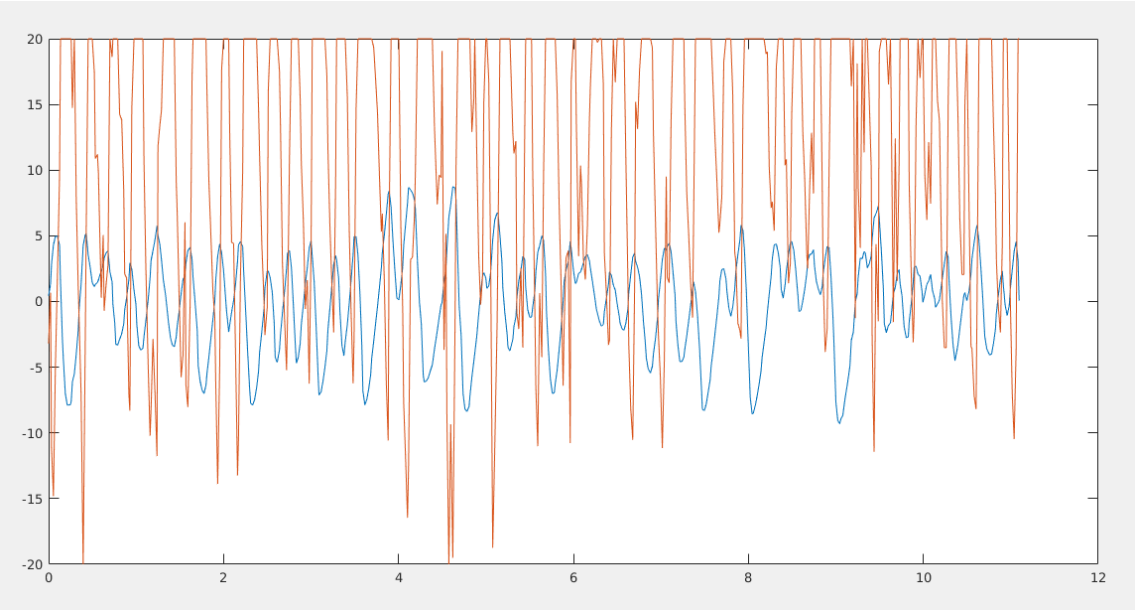


Figura 2.6: Gràfica d'entrada i sortida del sistema δ - pitch

$$G(s) = \frac{0,4812 - 8,879s}{s^2 + 19,09s + 89,31}$$

En la següent figura (figura 2.7) mostra la funció de transferència i el sistema real responent a la senyal d'entrada anterior:

Un cop aquí s'ha aplicat un anàlisis freqüencial(figura 2.8) per a tornar a aplicar el mètode de Philips.

Al dissenyar un PID pel mètode de Phillips s'ha de definir un angle de marge de fase per al PID i una pulsació de tall (ω_t), com més alta sigui aquesta, més ràpid serà el PID. El problema és que amb una rapidesa gaire alta el PID no és capaç de controlar el sistema, així doncs s'escull un marge de 60 graus i una pulsació de tall de 10rad/s per tenir una resposta acceptable. Amb aquests valors en ment i mirant a la gràfica, els valors més pròxims a

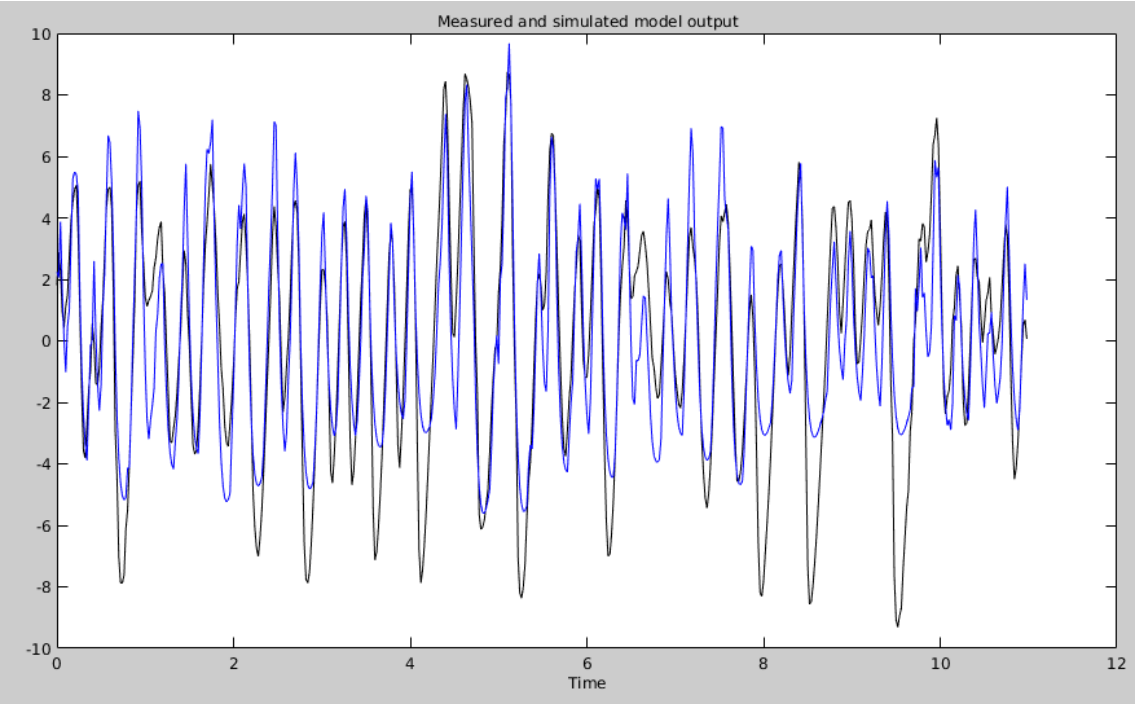


Figura 2.7: Resposta del model i del sistema real al senyal d'entrada

10rad/s que trobem són a 9,82rad/s, i corresponen a un guany de -6,65dB i un angle de fase de 178 graus. Així doncs l'angle total del PID (θ_{PID} és:

$$\theta_{PID} = 60° - (180° - 178°) = 62°$$

I degut a que el guany del PID per la planta ha de ser 1, es troba que el guany del PID ha de ser:

$$|PID| = \frac{1}{0,46505} = 2,15$$

A partir d'aquí, com que la Kp és la part real del PID, i com que se'n coneix el mòdul i l'angle es pot trobar de forma senzilla:

$$Kp = 2,15\cos(62°) = 1,0095$$

Per altre banda:

$$PID = Kp(1 + \frac{1}{Tis} + Tds)$$

i com que s= ω j arribem a:

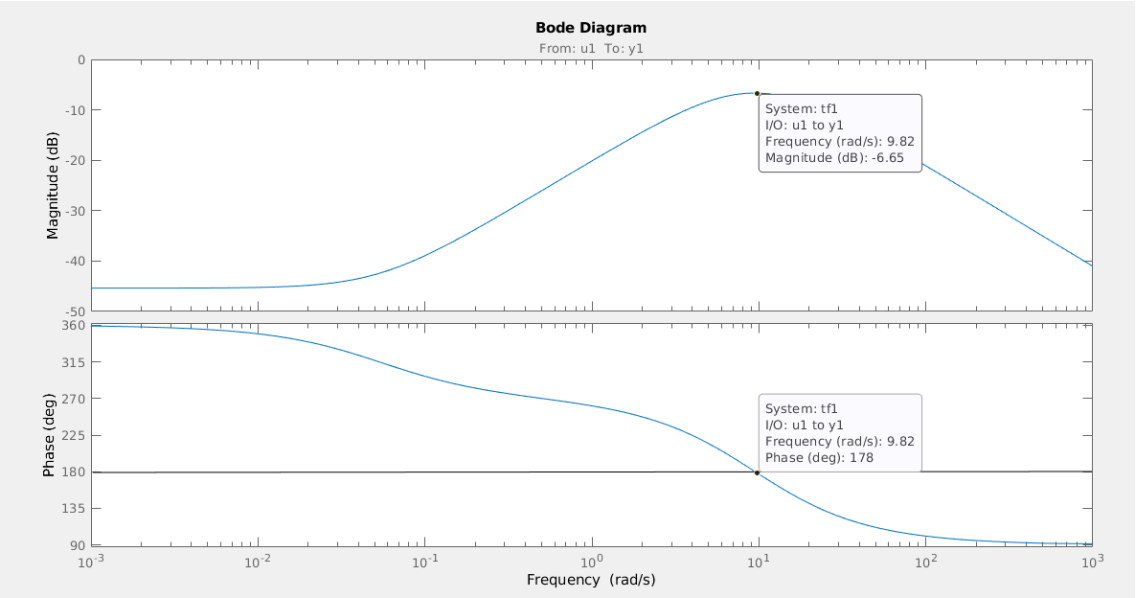


Figura 2.8: Anàlisis freqüencial

$$\theta_{PID} = \arctan(Td\omega - \frac{1}{Ti\omega})$$

que per a la pulsació de tall escollida (ω t = 10 rad/s):

$$tan(62°) = 10Td - \frac{1}{10Ti}$$

Per a poder resoldre l'equació anterior manca una equació, així que s'usa *Ziegler-Nichols* [5], que estipula que un bon valor de Ti és quatre vegades Td, així doncs s'arriba a:

$$10Td - \frac{1}{4Td10} - \tan(62°) = 0$$

de la qual s'obté que Td = 0,2042, i que per tant:

$$Kd = Kp\Delta Td = 1,0095\Delta 0,2042 = 0,2061399$$

$$Ki = \frac{Kp}{Ti} = \frac{1,0095}{4\Delta 0,2042} = 1,2359$$

així doncs el PID dissenyat és 2.3:

Kp	Ki	Kd
1,0095	1,2359	0,2061399

Taula 2.3: Taula de les constants del PID inicial

Un cop aquí s'ha realitzat un experiment amb el robot aplicant el PID dissenyat (filtrant la sortida del regulador derivatiu amb un filtre exponencial degut a que el feia inestable)i s'ha obtingut el següent resultat (limitant la sortida màxima del PID a 20 graus):

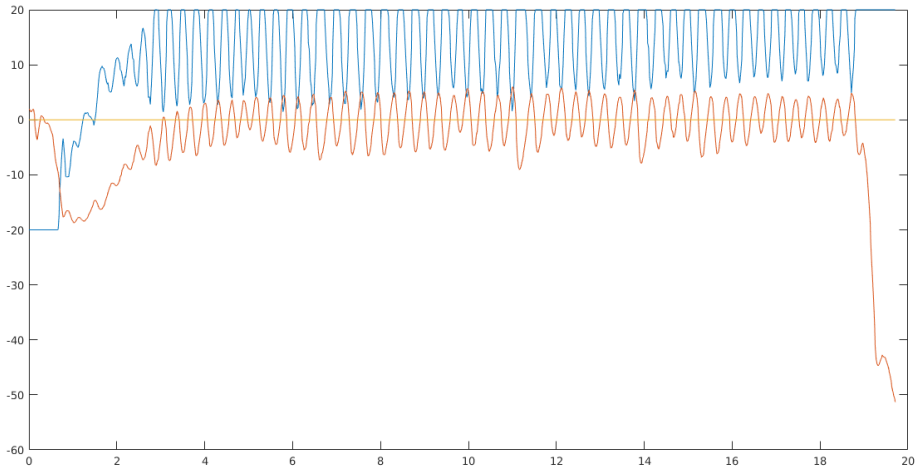


Figura 2.9: Senyal i resposta del PID teòric

S'aprecia que el PID es satura i que oscil·la molt, així doncs, després de fer un ajust fi de les constants (provant de modificar les constants una a una fins a aconseguir un resultat satisfactori) i augmentant el límit de la sortida a 30 graus s'ha arribat a un resultat més plaent 2.10:

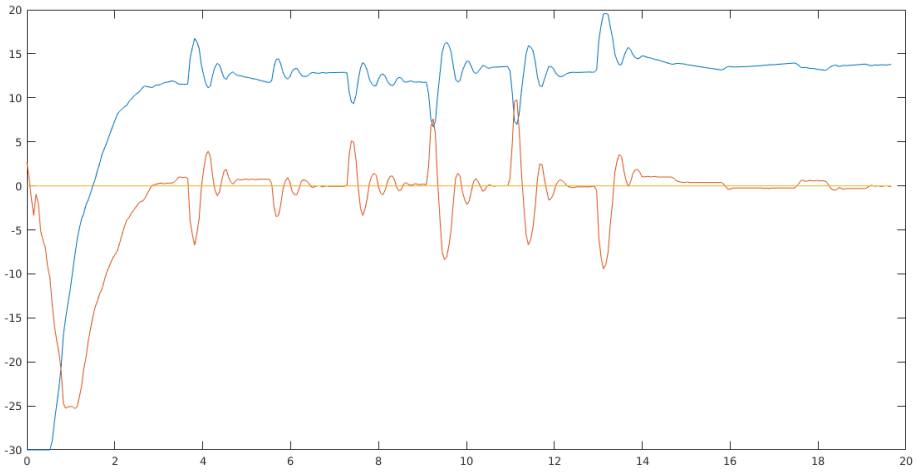


Figura 2.10: Senyal i resposta del PID ajustat

En aquesta gràfica es pot apreciar que el PID no només estabilitza el sistema sinó que també té una bona resposta en front les pertorbacions. Els valors de les constants són els esmentats a continuació:

Kp	**Ki**	**Kd**
0,15	1,76	0,19

Taula 2.4: Taula de les constants del PID ajustat

Pel que fa a la resta d'articulacions (angles γ i β) s'han assajat a factors pels quals es multiplica la sortida del PID per aconseguir una resposta idònia (corresponent al bloc de factors estabilitzadors), així doncs s'ha arribat als valors descrits en la taula següent:

factor δ	**factor** γ	**factor** β
1	0,5	0,125

Taula 2.5: Taula dels factors estabilitzadors

Amb aquests paràmetres s'ha arribat a el resultat final mostrat a la figura 2.11, on es pot apreciar que les pertorbacions són ben corregides, i tot i que

l'assoliment de la posició desitjada és lenta, el robot és molt estable i robust.

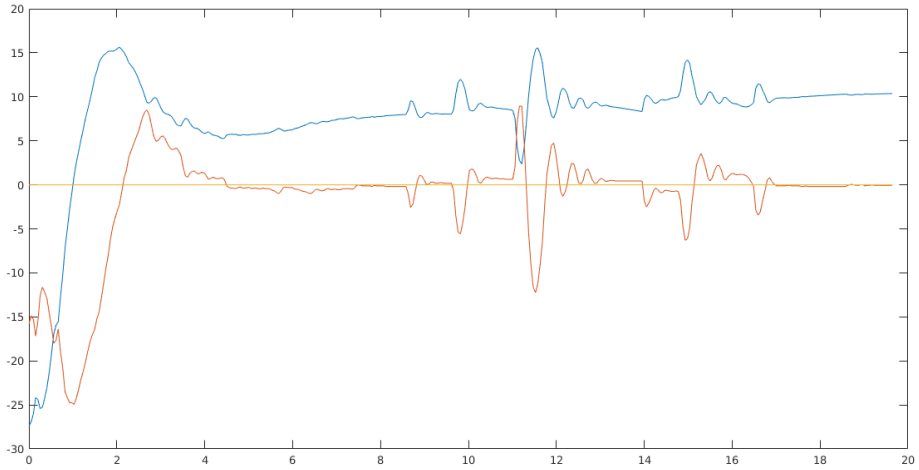


Figura 2.11: Senyal i resposta del PID amb factors

Cinemàtica inversa: la cinemàtica inversa es pot entendre com el conjunt d'expressions que proporcionen els angles que s'ha de proporcionar a cada articulació per assolir la posició desitjada. En aquest apartat es mostrarà el càlcul de la mateixa per a les cames d'en Tamtam.

A la figura següent(2.12) es mostra un esquema amb la cama dreta projectada en els plans **XZ** i **YZ** i s'hi defineixen tots els angles i distancies necessàries per a poder-la calcular. **T** està referit respecte **F** i té les coordenades (**X**, **Y**, **Z**).

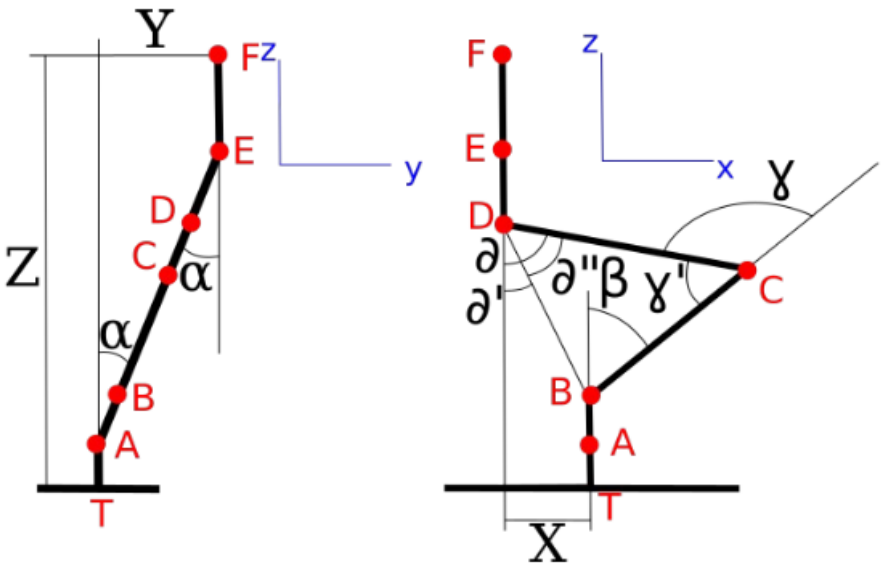


Figura 2.12: Vista general de la cama dreta

Així doncs, el càlcul de α és directe i esdevé:

$$\alpha = \arctan(\frac{Y}{Z - \overline{TA} - \overline{EF}})$$

A partir de α es pot calcular la distància entre B i D usant pitàgores:

$$\overline{BD} = \sqrt{X^2 + (Y - \overline{AB}\sin\alpha - \overline{DE}\sin\alpha)^2 + (Z - \overline{AB}\cos\alpha - \overline{DE}\cos\alpha - \overline{TA} - \overline{EF})^2}$$

I així s'obté un triangle del qual se'n coneixen totes les distancies però no els angles.

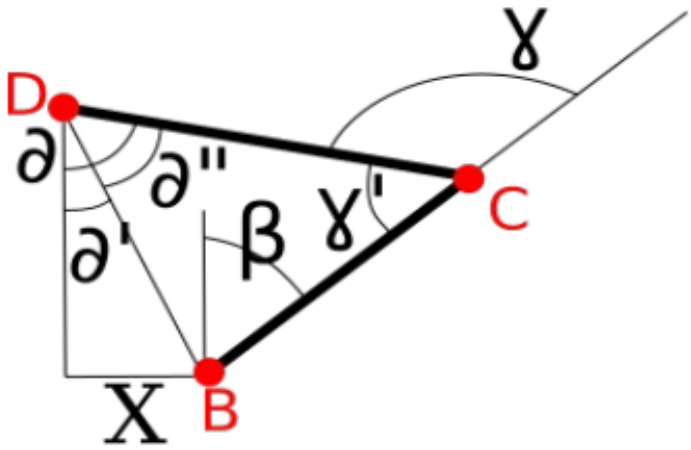


Figura 2.13: Vista del triangle BCD

Aplicant el teorema del cosinus s'obté γ':

$$\gamma' = \arccos(\frac{\overline{BC}^2 + \overline{CD}^2 - \overline{BD}^2}{2\overline{BCCD}})$$

i com que γ=180-γ' :

$$\gamma = 180 - \arccos(\frac{\overline{BC}^2 + \overline{CD}^2 - \overline{BD}^2}{2\overline{BCCD}})$$

també es pot trobar δ ' :

$$\delta' = \arcsin(\frac{X}{\overline{BD}})$$

I δ":

$$\delta'' = \arccos(\frac{\overline{CD}^2 + \overline{BD}^2 - \overline{BC}^2}{2\overline{CDBD}})$$

Com que δ, ja que $\delta = \delta$ ' + δ":

$$\delta = \arcsin(\frac{X}{\overline{BD}}) + \arccos(\frac{\overline{CD}^2 + \overline{BD}^2 - \overline{BC}^2}{2\overline{CDBD}})$$

Per últim, com que la suma dels angles interns d'un triangle sempre sumen 180:

$$\beta = 180 - \delta - (180 - \gamma) = \gamma - \beta$$

Calcul de trajectòria Aquest és l'ultim bloc de l'esquema, tracta d'un bloc que agafa la posició dels obstacles a partir de la càmera i la posició de la línia de meta. Un cop calculada la trajectòria va generant punts els quals el robot ha de seguir, i si es desvia de la trajectòria estipulada, el *trace pad* ha de tornar a calcular-la per a redirigir l'humanoide en la bona direcció.

De moment aquesta part sols compta amb la generació inicial de la trajectòria. Per al seu funcionament es precisa el coneixement de la posició dels d'obstacles i la del robot. Funciona dividint l'espai (de moment limitat a 250cm x 200cm) en una quadrícula de 25cm de costat. A partir d'aquí hi disposa els obstacles de tal forma que ocupin una o varies caselles i es posiciona ell, creant una matriu en la qual cada casella correspon a un quadre de la quadrícula. Un cop aquí guarda la posició dels obstacles i la posició del propi robot a la matriu i ho mostra per pantalla (figura 2.14).

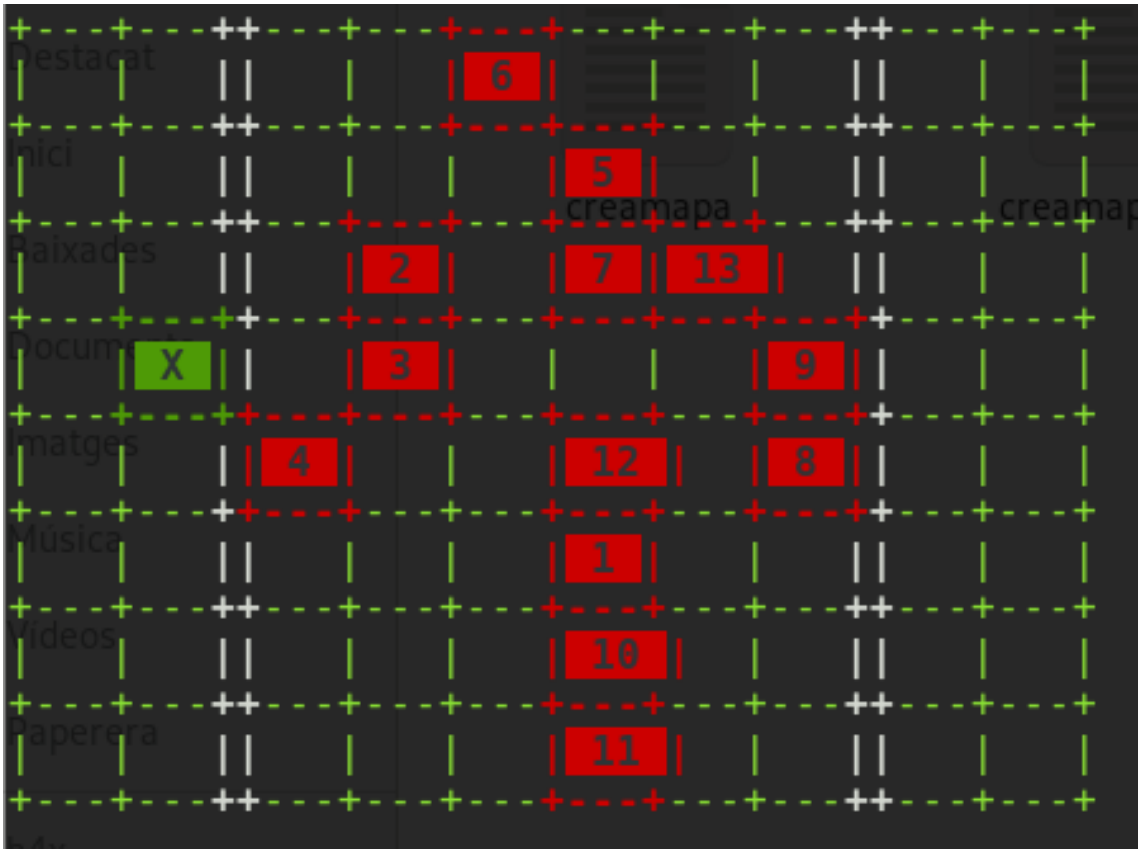


Figura 2.14: Mapa generat a partir de la ubicació dels QR i del robot

A partir d'aquí es crea un vector de mida variable que guardarà les coordenades de la casella. Primer comprova les quatre caselles que toquen la del robot. Les que són obstacle, paret, o ja estan afegides al vector són descartades, les altres són afegides al vector juntament amb una variable que indica a quina casella pertany la iteració. Després es va repetint per a cada element del vector fins que s'arriba a el lloc desitjat o s'arriba al final del mateix. Si s'arriba al final del vector el programa detecta que no hi ha trajectòria possible, així que és descartat. Si per contra el programa troba el final del camí, es ressegueix de forma inversa l'itinerari seguit i s'arriba a la trajectòria més curta que ha de seguir per arribar a la meta (2.15).

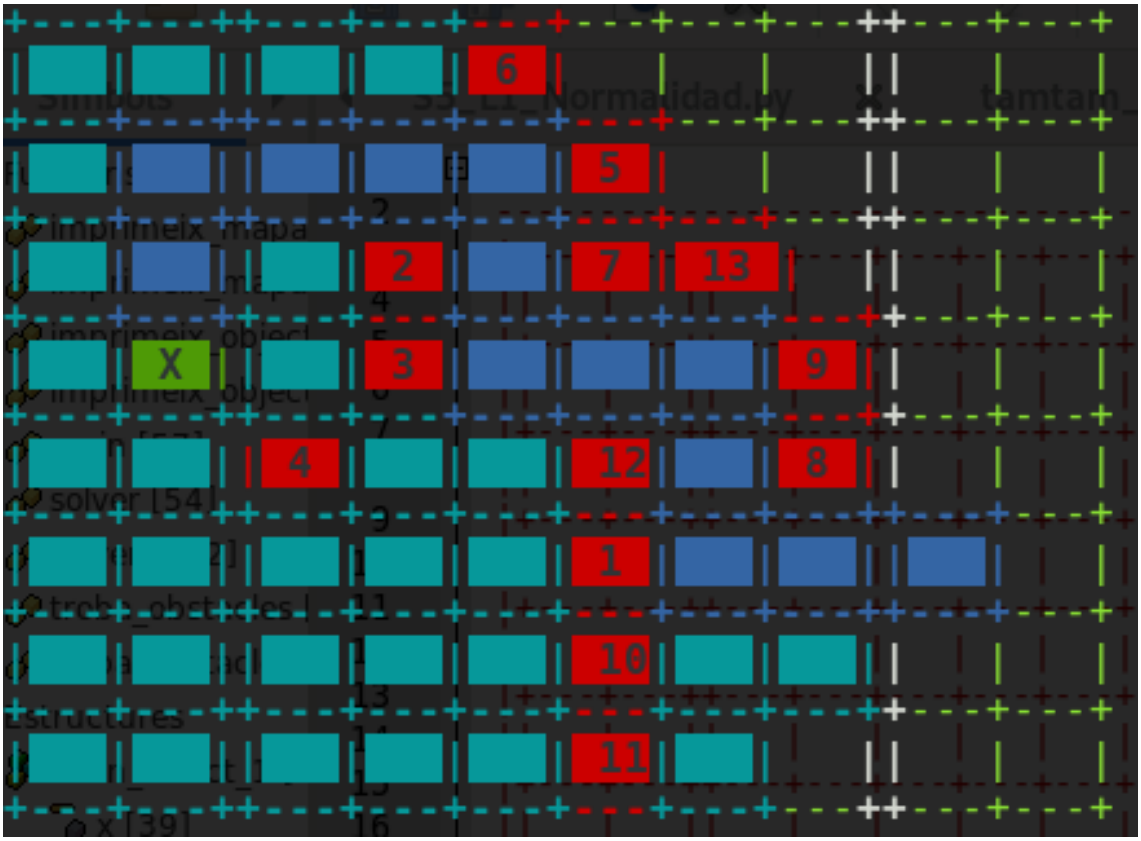


Figura 2.15: Mapa amb la trajectòria generada

Aquest mètode és molt lent i ineficient, tanmateix és una solució funcional que permet el desenvolupament de la resta de parts.

2.2.3 Software pel control de l'humanoide

En aquest apartat es descriurà el disseny del conjunt de programes que faran que l'humanoide funcioni. Per començar, es programarà l'ATMEGA per ser esclau I2C i canviar el color dels LEDs RGB a voluntat. També se li podrà consultar el nivell de la bateria, ja que és ell el que compta amb les úniques entrades analògiques que hi haurà. Després s'ha de fer una llibreria en C++ per al control del robot. El C++ és un llenguatge de programació d'alt nivell que deriva del C. Aquest llenguatge és dels més emparats, ja que és molt ràpid, potent i permet l'ús d'objectes. Així doncs es farà servir per programar els codis que regularan el comportament de l'humanoide. Es proposa la creació d'una llibreria que defineixi un objecte (Tamtam) amb tots els atributs i mètodes necessaris per al seu control.

La programació de la visió artificial es farà amb l'ús de les llibreries OpenCV, unes llibreries *open source* (de software lliure) que permeten el tractament d'imatges i l'extracció d'informació molt valuosa. Aquestes llibreries estan implementades en c, c++ , python i Java, i suporten els sistemes operatius de Linux, MacOS i Windows, per tant tenen molta versatilitat.

Aquestes llibreries es combinaran amb zbar i s'encarregaran de calcular la ubicació dels QR respecte del robot a partir de la seva posició a la imatge. Aquests QR estan muntats sobre una base de cartró que els recolza. La forma de posicionar els QR és fent un canvi de base de les coordenades de la imatge a les coordenades reals. Per a fer això es pot fer us d'una transformada homogènia, tanmateix abans s'ha de corregir la imatge. La correcció de la imatge és necessària degut a que la lent que porta distorsiona la seva captació, és per això que es defineix els paràmetres intrínsecs de la càmera com els paràmetres necessaris per corregir aquesta distorsió. L'equació que s'usarà és la següent[3]:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

En aquest document es definirà l'anterior expressió de forma reduïda :

$$\lambda I = KHtP$$

On I és la matriu que expressa la coordenada d'un píxel a la imatge en les coordenades de la imatge. L'origen de coordenades està a la part superior esquerre i l'eix d'abscisses (coordenada x o u) va d'esquerre a dreta, metre

que l'eix de les ordenades (y o v) va de dalt a baix.

La matriu K és la matriu dels paràmetres intrínsecs de la càmera. Els seus paràmetres són (c_x,c_y), que expressen el punt principal (el centre real de la imatge) en píxels i f_x i f_y que corresponen a les distàncies focals en píxels. Aquesta matriu depèn únicament de la càmera, pel que un cop calibrada ja no és necessari tornar-ho a fer.

Ht és la matriu de la transformada homogènia de la coordenada real a la coordenada a la imatge (però sense la última fila [0, 0, 0, 1], i el que fa és transformar un punt del qual se'n coneixen les coordenades reals a la seva coordenada dins la imatge.

P correspon al punt real en la seva forma homogènia.

Així doncs, es necessita una calibració de la càmera per a poder trobar la matriu dels paràmetres intrínsecs. Aquesta calibració s'ha dut a terme amb un taulell d'escacs i un programa escrit en python que és el que duia a terme, el qual s'ha modificat d'un altre ja existent [2]. Per a poder-ho fer primer s'han fet fotografies amb el taulell d'escacs en diferents posicions i orientacions de tal forma que quedés representat a tantes zones de la imatge com ha estat possible. S'ha fet un total de 25 imatges i s'han guardat en la mateixa carpeta on es troba l'script. Al fer córrer l'script s'ha arribat a això:

$$\text{K}= \begin{bmatrix} 2329,8 & 0 & 1332 \\ 0 & 2363,7 & 950 \\ 0 & 0 & 1 \end{bmatrix}$$

Pel que fa a la matriu *Ht* surt de forma directe ja que és sols una rotació, i el resultat és el següent:

$$\text{Ht}= \begin{bmatrix} 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

així doncs, si es desenvolupa s'arriba a:

$$\begin{gathered} \lambda u = X c_x - Y f_x \\ \lambda v = X c_y - Z f_y \\ \lambda = X \end{gathered}$$

Com que en aquest cas el que es desitja és obtenir el punt real a partir del punt de la imatge s'ha d'aïllar X, Y i Z i resoldre el sistema d'equacions. El problema és que hi ha 4 incògnites, ja que λ també és desconeguda, tanmateix es pot conèixer la Z del punt real, ja que els cartells amb els QR tenen mides conegudes, i el centre del QR es troba a una Z absoluta de 160mm, tanmateix la càmera es troba a 185 mm, el que fa que:

$$Z = 160 - 185 = -25$$

Amb aquestes dades i desenvolupant s'arriba a:

$$\lambda = \frac{Z f_y}{c_y - v}$$
$$X = \lambda$$
$$Y = \frac{X c_x - \lambda u}{f_x}$$

on Z és l'altura del punt referit a la càmera.

2.2.4 Fonts d'alimentació

Un dels problemes crítics en un projecte com aquest és el d'escollir les fonts d'alimentació, ja que és una de les parts més pesants del robot i també una de les delicades. En Tamtam usa actuadors que requereixen 6 volts, mentre que l'electrònica necessita 5 volts per a poder funcionar. Una de les possibles solucions és alimentar els motors a 5 volts i buscar una bateria que sigui capaç d'alimentar als motors i a l'electrònica. En aquest cas s'ha decantat per l'ús d'una *LiPo*.

Les *LiPo*s són unes bateries confeccionades amb polímers i liti molt usades en robòtica degut a la que tenen una alta capacitat, poden donar uns pics molt alts de descàrrega i també tenen un pes lleuger. Aquestes bateries estan compostes de cel·les que tenen un voltatge màxim de 4,2 volts i es poden arribar a descarregar fins als 3 volts. Mai s'ha de baixar dels 3 volts per cela, sinó la bateria ja és irrecuperable i s'ha de descartar.

En aquest projecte s'ha escollit fer servir dues bateries de dues cel·les (de 8,4 volts a 6 volts) amb una capacitat de 1000 mAh cadascuna i ajuntar-les en paral·lel per tal de doblar la capacitat, i per tant doblar l'autonomia del robot. Estaran disposades a la part frontal del robot (dins la tapa frontal del

cos) i serviran per alimentar l'electrònica i els servomotors.

Per transformar el voltatge de la LiPo a un voltatge adequat per als motors i a l'electrònica es farà us dels convertidors *Buck*, que són dispositius electrònics que redueixen el voltatge d'una font d'alimentació a un altre voltatge per a la càrrega conservant-ne la potencia. Per a aconseguir el resultat desitjat es farà us d'un convertidor *Buck* a 5 volts sempre que a l'entrada s'hi subministrin més de 6 volts i un altre a 6 volts, que els subministrarà sempre que a l'entrada hi hagi més de 7,2 volts. Aquests valors depenen de cada models i de cada fabricant. A la figura 2.16 es pot apreciar de forma esquemàtica com estarà disposat. Entre el positiu de la bateria i la potencia hi haurà un interruptor.

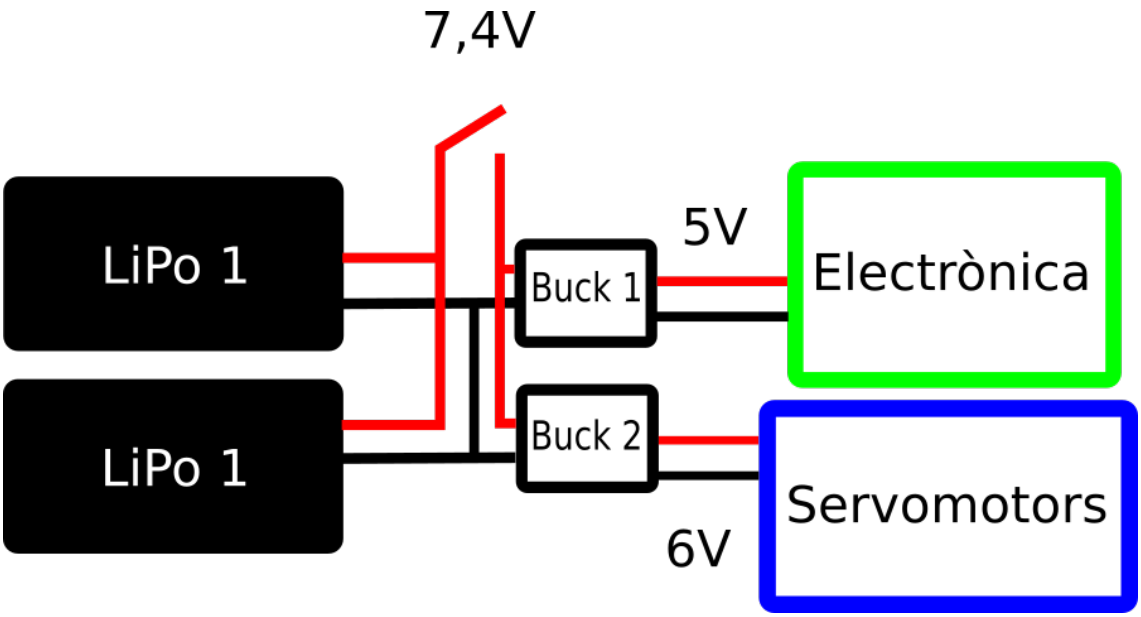


Figura 2.16: Esquema de l'alimentació

2.2.5 Electrònica

A l'hora de dissenyar l'electrònica s'ha de tenir en compte tots els elements que hi haurà i com s'han de connectar. Per començar tenim tres dispositius que van en un bus I2C, i que per al seu bon funcionament es recomana posar una resistència de 4K7Ω entre l' SDA i els 5V, també es recomana el mateix per al l'SCL, i serà implementat en la pròpia placa. A més a més els dos dispositius esclaus que pengen del bus (l'SD20 i l'Atmega328) funcionen a

5 volts, mentre que el dispositiu mestre (la Raspberry Pi Zero) s'alimenta a 5 volts però funciona a 3V3. Això implica que necessitem un convertidor bidireccional de 5 volts a 3,3 volts entre els esclaus i la Raspberry. Pel que fa a la IMU es pot alimentar entre 3,3 volts i 5 volts. Si s'alimenta a 3,3 volts ja no cal convertidor entre la IMU i la Raspberry, per sort la Raspberry compta amb sortides de voltatge de 3,3 volts dels quals es poden agafar l'alimentació. La IMU s'ha de preparar per a poder treballar via serie, l'esquema de connexions és el següent:

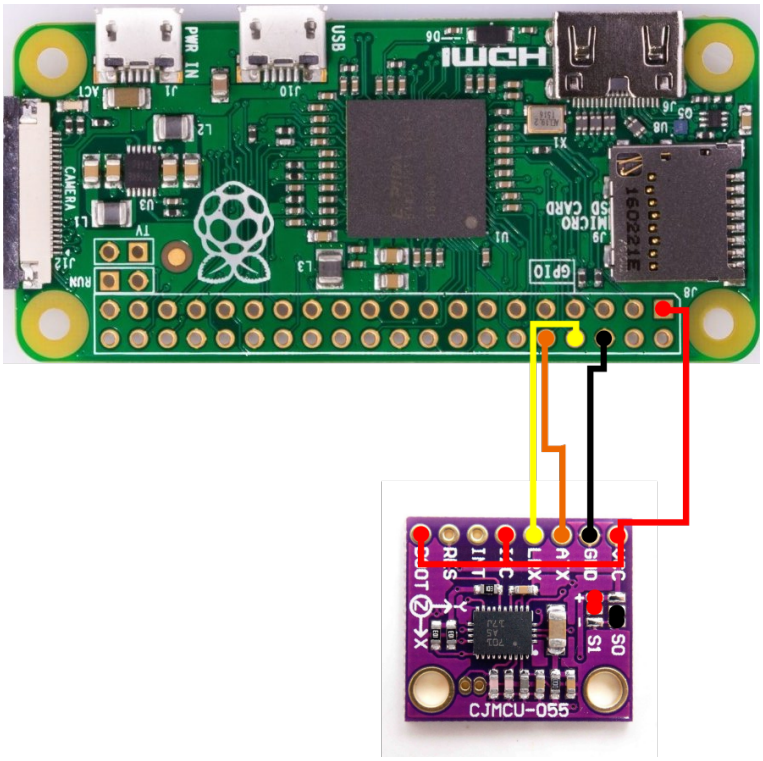
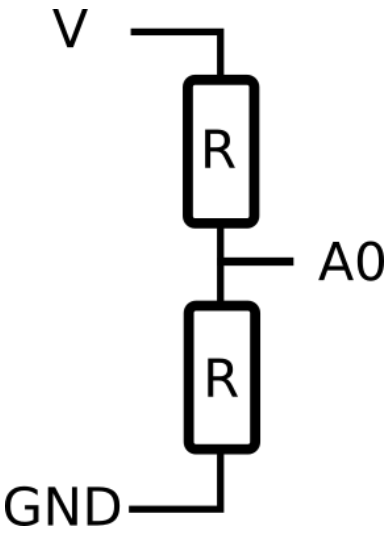


Figura 2.17: Esquema de connexió de la IMU a la Raspberry
[8] [4]

Per altre banda la Atmega ha de tenir una entrada analògica que connecti amb l'alimentació, sinó no es podrà mesurar el nivell del voltatge de la bateria. Aquesta connexió no pot ser directe, ja que el voltatge màxim que suporta l' Atmega és de poc més de 5 volts, i el voltatge mínim que pot donar la bateria per al bon funcionament del robot (afegint-hi una mica de marge) és de 7,5 volts, per tant es proposa un divisor de voltatge per a dividir el voltatge a la meitat. Definida I com la intensitat que passa de V a GND:



$I = \frac{V}{2R}$
$V_{A0} = IR = \frac{V}{2R}R = \frac{V}{2}$

Figura 2.18: Esquema del divisor de voltatge

D'aquesta forma s'aconsegueix la meitat del valor del voltatge de la bateria. Per a llegir-ne el nivell sols fa falta una regla de tres, ja que és una mesura lineal i es coneix que el voltatge màxim està en 5 volts i correspon al valor 1023 (ja que l'entrada analògica té una resolució de 10 bits ($2^{10} = 1024$, però com que també es considera el 0 és $2^{10} - 1 = 1022$). A partir d'aquí es pot definir el valor 1023 com a equivalent a 10 volts (ja que s'ha reduït el voltatge a la meitat amb el divisor de voltatge, i d'aquesta forma es coneixerà el voltatge real). Així doncs:

$$Val_{max} = 8,4\frac{1023}{10} = 859$$

$$Val_{min} = 7,5\frac{1023}{10} = 767$$

Ara ja es coneixen els valors corresponents al màxim i al mínim del nivell de bateria amb una precisió de 9,78mV. Per altre banda, degut a que el sistema de comunicació amb la Raspberry és per I2C, i que aquest usa paraules de 8 bits (paquets d'informació de 8 bits) s'ha de passar el valor obtingut de 10 bits a 8 bits. La forma més fàcil de fer això és fer un desplaçament a la dreta de dos bits, el que implica perdre'n la seva resolució, tanmateix obtenim un error màxim de 39mV, més que acceptable per aquest projecte. Així doncs, fer un desplaçament a la dreta de dos bits equival a dividir per 4 el resultat anterior:

$$Val_{max} = \frac{859}{4} = 214$$

$$Val_{min} = \frac{767}{4} = 191$$

Aquests correspondran als valors limit que llegirà la Raspberry.

S'ha de recordar també que la placa ha de comptar amb connexions auxiliars per si en un futur es vol estendre la seva funcionalitat. Així doncs s'ha dissenyat la placa i ha quedat de la següent forma:

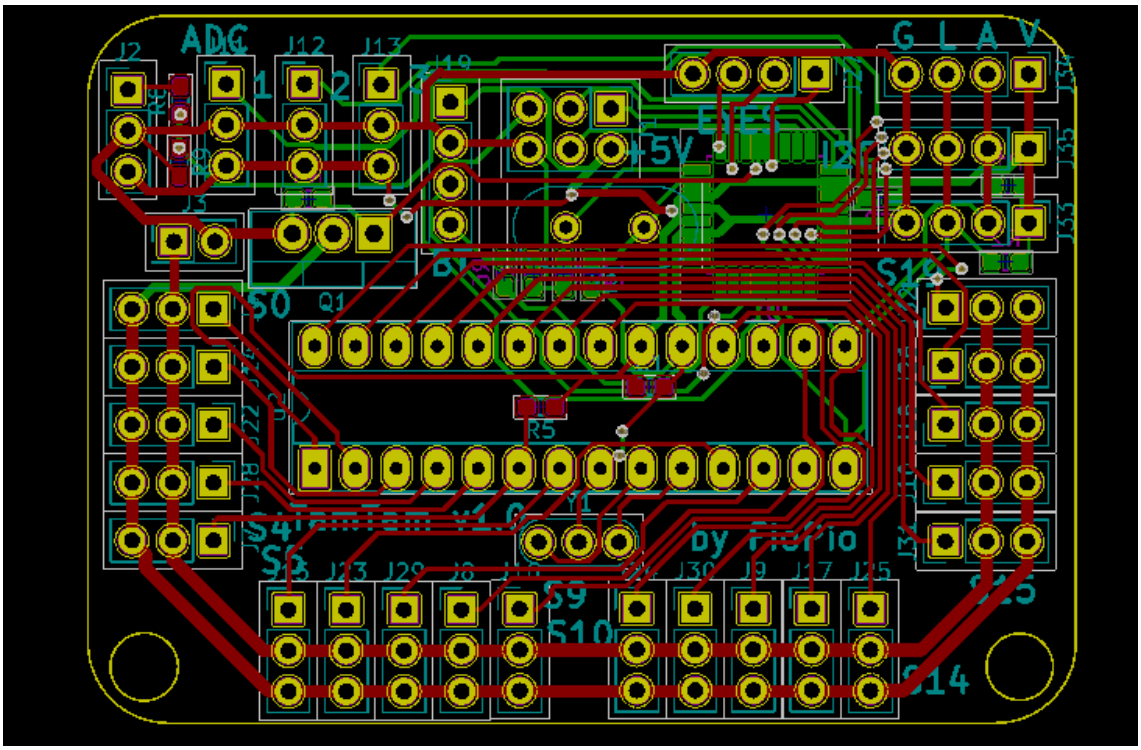


Figura 2.19: Disseny de la placa

Un cop fet el disseny de la placa, s'ha exportat a gerber (extensió per a poder-la fabricar) i s'ha enviat a elecrow a fer (una empresa que fa plaques via web). A continuació es mostra la placa amb una breu explicació de les connexions:

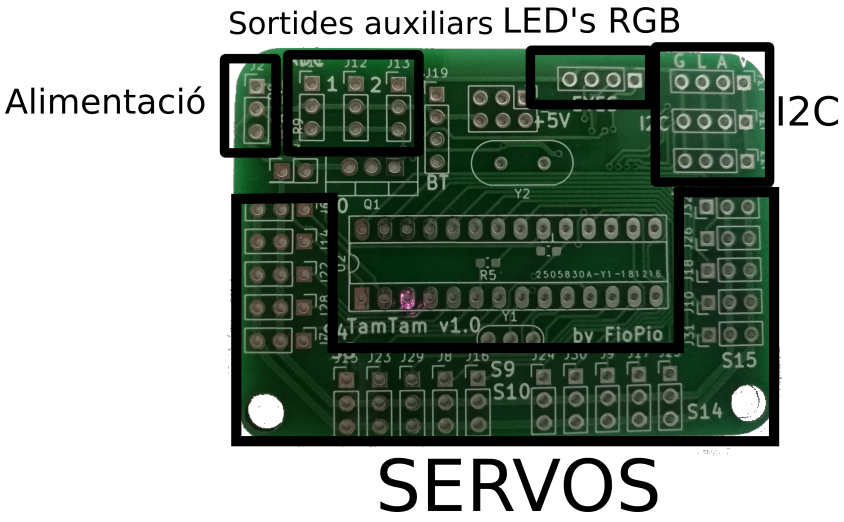


Figura 2.20: Placa i sortides

Per acabar s'ha soldat i comprovat que funcionés tot bé.

Figura 2.21: Placa soldada cara anterior

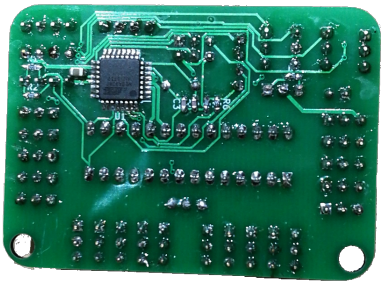


Figura 2.22: Placa soldada cara posterior

2.3 Desenvolupament i implementació

Un cop el hardware està funcionant cal implementar tot el que s'ha esmentat anteriorment. El problema més gran que hi ha hagut ha estat el de la lectura dels valors de la IMU, ja que es tenien les llibreries per a Arduino i per a python, però no per C++, i això ha estat un problema perquè s'ha estat tres setmanes intentant fer-ne una adaptació del C. Per desgracia cap dels intents ha estat fructífer, així que s'ha optat per usar el codi en python [9] i barrejar-lo amb la llibreria de C++ dissenyada usant ROS. ROS (*Robotic Operating System*) és una capa extra sobre el sistema que fa córrer diversos programes en paral·lel (anomenats nodes) i permet l'intercanvi d'informació entre ells (anomenats missatges). La idea de ROS és dividir un programa complex en nodes més petits que facin coses senzilles i intercanviïn informació. D'aquesta forma es pot tenir un node programat en python que llegeixi l'estat de la IMU i ho publiqui en un tòpic (lloc on s'envien els missatges i des del que els altres nodes els poden llegir). El node principal rebrà la informació publicada en aquest tòpic i la usarà per al càlcul del PID. Paral·lelament, el node que s'encarregui de calcular la posició dels QR la publicarà en un altre tòpic. I així el node que s'encarregui del tracepad podrà obtenir la orientació i la posició dels obstacles, per a enviar un altre missatge al node principal de què ha de fer. A la figura 2.23 es pot veure un esquema de la seva disposició.

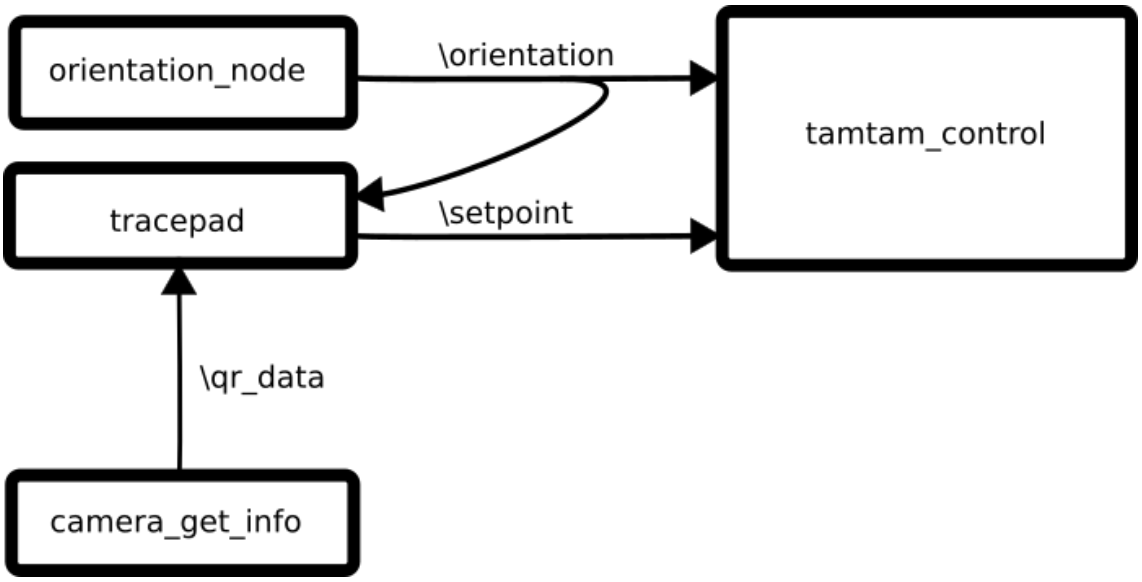


Figura 2.23: Esquema nodes i tòpics

2.3.1 Nodes

tamtam_control Aquest node està subscrit als tòpics *orientation* i *setpoint*, i s'encarrega de calcular la consigna que s'ha d'enviar a les articulacions. A més a més, com que les comunicacions via I2C només les pot fer un node a la vegada, aquest també s'encarrega del control de la informació que s'envia o es rep per aquest canal. Així doncs aquest node rep informació de la seva orientació i a on ha d'anar i envia la senyal als motors que correspon per arribar a aquesta posició. També s'encarrega de llegir l'estat de la bateria i de definir el color dels ulls (fent-los servir com a indicadors).

orientation_node Aquest node està basat en el codi proporcionat per *Adafruit* [9] i s'ha adaptat per a ser un node de ROS. El que fa és llegir el pitch , roll i yaw de la IMU cada 20 mil·lisegons i el publica al tòpic *orientation*.

tracepad Aquest node rep informació dels tòpics *orientation* i *qr_data*, calcula la trajectòria i intenta corregir les desviacions. Gràcies a la informació d'aquests dos tòpics pot trobar la seva orientació i la variació del punt anterior i corregir el rumb. Publica al tòpic *setpoint*, que descriu la posició del robot desitjada.

camera_get_info Aquest node fa una foto i obté les coordenades de cada QR de la imatge en el món real referits al robot i el contingut del codi i ho publica al tòpic *qr_info*.

2.3.2 Tòpics

orientation Aquest tòpic usa un missatge del tipus *Orientation*, que és un missatge que consisteix en tres variables del tipus *float32* (un nombre decimal definit amb 32 bits) anomenades pitch, yaw i roll, i corresponen al pitch, yaw i roll del robot.

qr_data Aquest tòpic usa missatges del tipus *QR_data*, que contenen una variable de tipus *String* (cadena de caràcters amb un byte per a cada caràcter) anomenat data que conté la informació que porta el QR i dos *float32*

anomenats X i Y, que contenen les coordenades x i y del QR referent al robot en el món real.

setpoint Aquest tòpic fa servir missatges del tipus *Setpoint*, que conté dues variables del tipus *float32* anomenades dis i ang, i contenen la distància que ha de recórrer fins al següent punt i l'angle que ha de girar durant el trajecte.

2.3.3 Arbre de directoris

En aquest apartat es mostrarà com estan disposats els directoris dins la carpeta src del workspace de ros:

```
src
├── tamtam_control
│   ├── CMakeLists.txt
│   ├── package.xml
│   ├── src
│   │   ├── tamtam.cpp
│   │   ├── tamtam_control.cpp
│   │   └── tracepad.cpp
│   ├── include
│   │   └── tamtam.h
│   └── msg
│       └── Setpoint.msg
└── orientation
    ├── CMakeLists.txt
    ├── package.xml
    ├── src
    ├── include
    ├── scripts
    │   ├── orientation_node.py
    │   └── camera_get_info.py
    └── msg
        ├── Orientation.msg
        └── QR_data.msg
```

És important destacar que el primer node que s'ha d'arrencar és el de *orientation.py*, després el de *camera_get_info.py*, seguit del *tracepad* i per últim el *tamtam_control*.

Conclusions i treball futur

3.1 Discussió de resultats

Un cop acabat el projecte s'ha de discutir si s'ha arribat als objectius definits al seu inici. Aquests objectius eren els següents:

- Estructura del robot funcional
- Electrònica del robot funcional
- Estabilitzar el robot
- Fer caminar el robot de forma estable
- Trobar els codis QR en la imatge
- Ubicar els codis en l'espai
- Traçat de la trajectòria
- Seguiment de la trajectòria

Pel que fa a l'estructura del robot la principal dificultat ha estat en el seu disseny, ja que com que les peces s'havien d'imprimir amb l'ús d'una impressora 3D s'ha hagut de tenir en compte com funciona la impressora per aconseguir el seu millor resultat. Les peces encaixen perfectament i permeten els moviments del robot. A més a més en Tamtam té una estètica agradable, que es va buscar dins el possible. El robot mesura 225mm, pesa menys de 700g (exactament 653g) i compta amb 19 DOF.

Pel que fa a la seva electrònica treballa com s'esperava que ho fes, encaixa perfectament en els llocs on s'havia disposat per a ella i compta amb elements que en permeten la seva extensió en cas que més endavant fora necessari. Quan el consum dels motors és molt alt (quan fa moviments molt

extrems) el driver de motors falla i fa que els servomotors es desbordin, això es podria arreglar afegint un condensador entre el GND i VCC prop del servodriver (s'ha d'estudiar de quina capacitat) perquè evités que la caiguda de voltatge deguda al pic de corrent el deixés sense alimentació.

Després de molts experiments i simulacions s'ha arribat als paràmetres que conformaven un PID capaç d'equilibrar el robot fins i tot quan s'inclina la base sobre la qual està disposat. El sistema és una mica lent (si la pertorbació és important pot tardar fins a 2 segons a estabilitzar-se), però robust, ja que assoleix la consigna de forma precisa.

El robot és capaç de desplaçar-se caminant (tot i que faci us del lateral del peu per ajudar-se) de forma estable gràcies a una bona cinemàtica inversa calculada i a l'ús d'una màquina d'estats per fer el més simple i eficaç el seu moviment.

Degut a que la càmera no és una càmera USB i té una resolució molt alta, OpenCV no pot fer-ne un us adequat i per tant no és recomanable fer que en prengui les fotos. El que s'ha fet és que es cridi a una funció del sistema que si que ho fa bé i llegir la imatge des d'allà. Un cop aquí, ha estat relativament fàcil aconseguir que la llibreria Zbar trobés el QR i en donés el seu contingut.

Per a ubicar el QR en l'espai s'ha tingut algun problema i s'ha hagut de fer més d'un experiment, tant per trobar la matriu de paràmetres intrínsecs com per a poder trobar el conjunt de fórmules que en dedueixen la seva posició real. Tot i així els resultats obtinguts són satisfactoris.

Pel que fa al traçat de la trajectòria no s'ha disposat de prou temps per a perfeccionar-lo, així que es troba en un estat molt primitiu al qual li falta maduració (està limitat a un espai de 250cm x 200cm i usa un sistema de traçat de trajectòria molt rudimentari).

Per últim, el seguiment de trajectòries, degut a la cinemàtica inversa de l'humanoide, ha tingut una implementació directa.

3.2 Directrius de continuació del projecte

Per a una continuació del projecte es proposen els següents punts:

- **Estudi de la reducció de massa de l'humanoide** La reducció de

la massa de l'humanoide aconseguiria allargar l'autonomia del robot, per tant no requeriria de bateries tant grans i podrien ser més lleugeres, que tot i ser-ho ja bastant, són els elements més pesats del robot.

- **Correcció del problema de l'SD20 en pics de corrent dels servomotors** La correcció del problema de l'SD20 en pics es pot arreglar afegint un condensador com ja s'ha expressat anteriorment, sols queda estudiar quin i on (modificant la placa i tornar-la a fer o afegint-lo a la placa ja feta).

- **Millora del sistema actual de traçat de trajectòries** La millora del traçat de trajectòria es pot fer estimant altres paràmetres als de longitud (ara mateix busca el camí més curt). Pot per exemple, presentar camins més segurs, o més ràpids.

- **Disseny i implementació d'un sistema de correcció de trajectòria** Per a poder corregir la trajectòria s'espera que es pugui usar el node ja implementat *tracepad*.

- Afegir elements que la càmera pugui detectarEs poden afegir elements com rectangles, cercles i triangles de diferents colors que defineixin certes accions i que l'humanoide hagi de reaccionar a elles.

- **Adaptar mes el projecte a la normativa del CEABOT** Per acabar, no cal perdre de vista que aquest projecte s'ha plantejat com una simplificació del problema que presentava el CEABOT, és per això que és interessant el plantejament d'un escenari més similar al que es troba al concurs.

Bibliografia

[1] *Atlas | Boston Dynamics*. URL: https://www.bostondynamics.com/atlas (cons. 07-06-2019).

[2] *Camera Calibration — OpenCV-Python Tutorials 1 documentation*. URL: https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_calib3d/py_calibration/py_calibration.html (cons. 02-06-2019).

[3] *Camera Calibration and 3D Reconstruction — OpenCV 2.4.13.7 documentation*. URL: https://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html?highlight=calib (cons. 02-06-2019).

[4] Semaf Electronics. *9-DOF Absolute Orientation IMU Fusion Breakout - BNO055*. de. URL: https://electronics.semaf.at/9-DOF-Absolute-Orientation-IMU-Fusion-Breakout-BNO055 (cons. 08-06-2019).

[5] *Fractional Order Motion Controls*. en. Google-Books-ID: G_T_cZuLA2IC. John Wiley & Sons, des. de 2012. ISBN: 978-1-119-94455-3.

[6] Charles L. Phillips H. Troy Nagle Jr. *Sistemas de control digital*. 2a ed. Editorial Gustavo Gili, S.A., 1993. ISBN: 84-252-1335-5.

[7] *Honda Global | ASIMO*. URL: https://global.honda/innovation/robotics/ASIMO.html (cons. 07-06-2019).

[8] *Raspberry Pi Zero W*. URL: https://thepihut.com/products/raspberry-pi-zero-w (cons. 08-06-2019).

[9] *Software | BNO055 Absolute Orientation Sensor with Raspberry Pi & BeagleBone Black | Adafruit Learning System*. URL: https://learn.adafruit.com/bno055-absolute-orientation-sensor-with-raspberry-pi-and-beaglebone-black/software (cons. 08-06-2019).

[10] *System Identification Toolbox Documentation - MathWorks España*. URL: https://es.mathworks.com/help/ident/ (cons. 09-06-2019).

[11] James Vincent. *Twitter taught Microsoft's friendly AI chatbot to be a racist asshole in less than a day*. Març de 2016. URL: https://www.theverge.com/2016/3/24/11297050/tay-microsoft-chatbot-racist (cons. 07-06-2019).